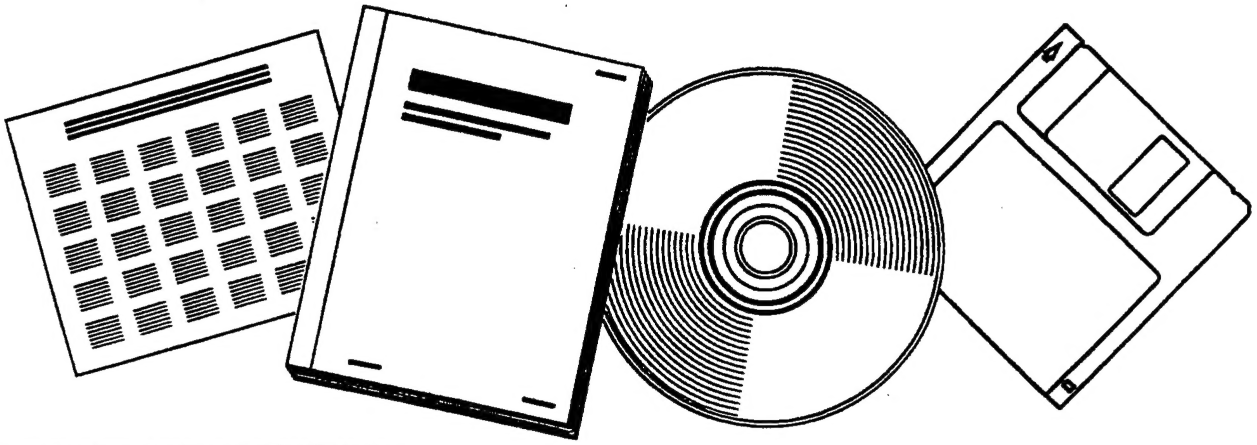
CONTINUOUS VERIFICATION BY DISCRETE REASONING

DEPARTMENT OF COMPUTER SCIENCE
STANFORD, CA

SEP 94



September 1994

Report No. STAN-CS-TR-94-1524

PB96-152764

Continuous Verification by Discrete Reasoning

by

Luca de Alfaro and Zohar Manna

Department of Computer Science

Stanford University

Stanford, California 94305

Continuous Verification by Discrete Reasoning

Luca de Alfaro and Zohar Manna*

Abstract

Two semantics are commonly used for the behavior of real-time and hybrid systems: a *discrete semantics*, in which the temporal evolution is represented as a sequence of snapshots describing the state of the system at certain times, and a *continuous semantics*, in which the temporal evolution is represented by a series of time intervals, and therefore corresponds more closely to the physical reality. Powerful verification rules are known for temporal logic formulas based on the discrete semantics.

This paper shows how to transfer the verification techniques of the discrete semantics to the continuous one. We show that if a temporal logic formula has the property of *finite variability*, its validity in the discrete semantics implies its validity in the continuous one. This leads to a verification method based on three components: verification rules for the discrete semantics, axioms about time, and some temporal reasoning to bring the results together. This approach enables the verification of properties of real-time and hybrid systems with respect to the continuous semantics.

*This research was supported in part by the National Science Foundation under grant CCR-92-23226, by the Defense Advanced Research Projects Agency under contract NAG2-892, and, by the United States Air Force Office of Scientific Research under contract F49620-93-1-0139.

1 Introduction

In order to use temporal logic to specify and verify properties of real-time and hybrid systems, some semantics must be chosen for the temporal behavior of the systems. There are two common choices [2, 18]. The first is a *continuous semantics*, in which the system evolution is represented by a series of time intervals, together with a mapping that associates to each point in time a state of the system. The second is a *discrete semantics*, in which the temporal evolution of the system is represented as an enumerable sequence of snapshots, each describing the state of the system at a certain time. Each of these semantics has its advantages and weaknesses.

The continuous semantics corresponds closely to the physical behavior of the system [8, 18]. System specifications describe the physical behavior, and therefore refer more directly to the continuous semantics than to the discrete one.

The discrete semantics enables the use of powerful verification rules to draw conclusions about the behavior of the system from premisses about its structure [6, 20]. The proof of the soundness of these rules depends in an essential way on the discreteness of the semantics, and in particular on reasoning by induction on the enumerable sequence of states. On the other hand, the discrete semantics corresponds less directly to the physical behavior of the system, and its relevance is in its relationship to the continuous semantics [8].

This paper shows that the advantages of the discrete semantics can be transferred to the continuous one. We show that if a temporal logic formula has the property of *finite variability*, its validity in the discrete semantics implies its validity in the continuous one. Most of the formulas that arise in practice have this property, and we give a series of simple criteria to characterize them.

This allows us to adapt the verification rules for temporal logic on the discrete semantics to the continuous one: if the conclusion of the verification rule is a formula with the finite variability property, it will also holds in the continuous semantics. In this way, we are spared the work of devising new verification rules for the continuous semantics.

We therefore propose a recipe for the verification of temporal logic properties of real-time and hybrid systems that consists of three ingredients: verification rules coming from the discrete semantics, axioms stating some basic properties of time, and a small amount of temporal reasoning to bring the two together. Temporal reasoning in the continuous semantics can be kept to a minimum, if desired.

In our representation, we follow closely the approach of [20], modeling real-time and hybrid systems by timed and phase transition systems respectively, and using a temporal logic containing both explicit time and age functions. As clocks are closely related to age functions, the results can be easily transferred to logics that use clocks as the basic timing construct.

We first present the case for real-time systems in some detail, and then we show the changes needed to adapt the results to hybrid systems.

2 Real-Time Systems

Real-time systems will be modeled by *timed transition systems* [7, 18]. A timed transition system $S = \langle \mathcal{V}, \Sigma, \Theta, \mathcal{T}, L, U \rangle$ consists of the following components.

1. A set $\mathcal{V}$ of variables called *state variables*, each with its type.
2. A set Σ of states: each state $s \in \Sigma$ is a type-consistent interpretation of all the variables in $\mathcal{V}$: we indicate with $s(x)$ the value at state s of x, for $x \in \mathcal{V}$.
3. A set $\Theta \subseteq \Sigma$ of initial states. Θ has an associated assertion $\Theta_f(\mathcal{V})$, such that $\Theta = \{s \mid s \models \Theta_f\}$, where s interprets $x \in \mathcal{V}$ as $s(x)$.
4. A set $\mathcal{T}$ of transitions, where $\tau \subseteq \Sigma \times \Sigma$ for all $\tau \in \mathcal{T}$. Each transition $\tau \in \mathcal{T}$ has an associated assertion $\rho_\tau(\mathcal{V}, \mathcal{V}')$ such that $\tau = \{(s, s') \mid (s, s') \models \rho_\tau\}$, where (s, s') interprets $x \in \mathcal{V}$ as $s(x)$ and x' as $s'(x)$.
5. Two sets L, U of minimum and maximum delays of transitions. For all $\tau \in \mathcal{T}$ it is $0 \leq l_\tau \leq u_\tau \leq \infty$.

We denote with c_τ the enabling condition of transition τ, defined by $c_\tau = \{s \mid \exists s'.(s, s') \in \tau\}$. For simplicity, we will assume that transitions are self-disabling: $(s, s') \in \tau \to s' \not\in c_\tau$.

The temporal behavior of a real-time system will be represented by *traces*. Corresponding to the discrete and the continuous views of the semantics, the formal representation of the behavior is given in terms of discrete and continuous traces.

2.1 Discrete Semantics

In the discrete semantics, each behavior is represented by a *discrete trace*, which is an enumerable sequence of *observations*. Each observation is a pair consisting of a snapshot of the system state and a timestamp indicating the time at which the snapshot was taken [8, 18, 7, 20].

Definition 1 (discrete trace) *A discrete trace σ_d is an enumerable sequence of observations $\langle s_0, t_0 \rangle, \langle s_1, t_1 \rangle, \langle s_2, t_2 \rangle, \ldots$, with $s_n \in \Sigma$, $t_n \in \mathbb{R}^+$ for $n \in \mathbb{N}$, such that*

$$t_0 = 0, \qquad \lim_{n \to \infty} t_n = \infty, \qquad \forall n \in \mathbb{N} : \; t_n \leq t_{n+1}.$$

A *position* of a trace is simply an integer $n \in \mathbb{N}$. If a trace represents a possible behavior of a system, we say that the system *admits* the trace.

Definition 2 (admission, discrete traces) *A timed transition system S admits a discrete trace σ_d: $\langle s_0, t_0 \rangle$, $\langle s_1, t_1 \rangle$, $\langle s_2, t_2 \rangle$, ..., written $S \triangleright \sigma_d$, if the following conditions are satisfied.*

1. *All the state changes are due to transitions that have been enabled at least for their minimum delay: for all $n \in \mathbb{N}$,*

$$s_n = s_{n+1} \vee \Big[t_n = t_{n+1} \wedge \exists \tau \in \mathcal{T} \Big[(s_n, s_{n+1}) \in \tau \wedge \forall k \big[k \leq n \wedge t_k > t_n - l_\tau \to s_k \in c_\tau\big]\Big]\Big].$$

2. Transitions are never enabled for longer than their maximum delay: for all $\tau \in \mathcal{T}$, $n, k \in \mathbb{N}$ with $k > n$,

$$t_k - t_n \leq u_\tau \vee \exists j \Big[n \leq j \leq k \wedge s_k \notin c_\tau\Big].$$

2.2 Continuous Semantics

In the continuous semantics, the behavior of the system is represented by a mapping from intervals of time to states of the system, and time is modelled by the set of real numbers. A trace is no more a sequence of snapshots, but a continuous representation of the evolution of the state of the system. Here, the word "continuous" is used in a different way than in calculus: it means that there are no gaps in the temporal description of the systems, such as the gaps between snapshots of the discrete semantics. It is this absence of gaps that makes the continuous semantics closer to physical reality.

Formally, a *continuous trace* is a sequence of pairs consisting of a state of the system and an interval of time spent by the system in that state. The intervals of time can overlap at most at the endpoints [8, 10, 2]. This semantics closely resembles the *superdense* semantics of [18]. If A is a linearly ordered set, we will indicate with Int_A the set of intervals (i.e. convex sets) of A.

Definition 3 (continuous trace) *A continuous trace σ_c is a sequence of pairs σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle$, ..., with $I_n \in \mathrm{Int}_{\mathbb{R}}$ and $r_n \in \Sigma$ for all $n \in \mathbb{N}$, such that:*

$$\forall n \Big(\sup I_n = \inf I_{n+1}\Big), \qquad \bigcup_{n \in \mathbb{N}} I_n = \mathbb{R}^+.$$

A continuous trace is closed if all its intervals I_0, I_1, I_2, ... are; it is open otherwise.

Definition 4 (moment) *A moment of a trace σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle$, ... is a pair (n, t) such that $t \in I_n$ [18]. The ordering $\leq$ of moments is the expected one:*

$$(n,t) \leq (n',t') \quad \textit{iff} \quad n < n' \vee (n = n' \wedge t \leq t').$$

In the following, when we write a pair (n, t) relative to a trace σ_c we will always assume that it is a moment of σ_c. We give the definition of admission only for closed traces. We define $I_n^{\leftarrow} = \inf I_n$, $I_n^{\rightarrow} = \sup I_n$. The definition of admission is then similar to the one given for discrete traces.

Definition 5 (admission, continuous traces) *We say that a timed transition system S admits a trace σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle$, ... if σ_c is closed, and the following conditions are satisfied.*

1. All the state changes are due to transitions that have been enabled at least for their minimum delay: for all $n \in \mathbb{N}$,

$$r_n = r_{n+1} \vee \exists \tau \in \mathcal{T}\Big[(r_n, r_{n+1}) \in \tau \wedge \forall k \Big[k \leq n \wedge I_k^{\rightarrow} > I_n^{\rightarrow} - l_\tau \rightarrow r_k \in c_\tau\Big]\Big].$$

2. Transitions are never enabled for longer than their maximum delay: for all $\tau \in \mathcal{T}$, $n, k \in \mathbb{N}$ with $k \geq n$,

$$I_k^{\rightarrow} - I_n^{\leftarrow} \leq u_\tau \vee \exists j \Big[n \leq j \leq k \wedge r_k \notin c_\tau\Big].$$

3 Temporal Logic

To express temporal properties of the behavior of the system, we use a multi-sorted temporal logic similar to the one proposed in [5, 6, 20].

Syntax. Our language contains flexible and rigid constants, rigid variables, rigid function symbols and predicates, the propositional connectives $\neg$, $\rightarrow$, the future temporal operators $\Box$, $\mathcal{U}$ and the past ones $\boxminus$, $\mathcal{S}$, and the symbols $=$ for equality and $\forall$ for quantification. From this basic set of symbols, additional ones can be defined as usual. Note that there is no next-time $\bigcirc$ operator in the logic.

The variables of the logic are *rigid*, meaning that they have the same value at all times; thus, quantification is allowed on rigid variables only [4]. The state variables of the system, whose value can change in time, are represented instead by flexible constants. This is different from the approach followed by [19], where quantification is allowed also on flexible variables, and where flexible variables (instead of flexible constants) are used to represent the state variables of the system. The approach followed here is such that a trace of the system will determine the model, and the variable assignment is used to deal with variables and quantification. To avoid confusion, for the rigid variables of the logic we use greek letters like ξ, ζ, and for the flexible state variables of the system latin ones like x, y.

Our language also contains the special flexible constant T of type *real*, whose value represents the time, and the interpreted predicate $<$ over the reals. Moreover, the language includes the *age function* Γ. For a formula ϕ, the term $\Gamma(\phi)$ indicates the length of the most recent interval in which ϕ has been continuously true [20]. We will assume that the argument ϕ of $\Gamma(\phi)$ does not contain occurrences T or nested age functions.

Semantics. The truth of temporal logic formulas is evaluated with respect to a model $\mathcal{M}$ and a variable assignment $\mathcal{I}$. A model $\mathcal{M} = \langle W, \leq, a\rangle$ is composed of a frame $\mathcal{F} = \langle W, \leq\rangle$ and an assignment function a. The frame is a set W of *worlds* together with a relation of reflexive linear order $\leq$. Each world represents an instant of time, and the order relation $\leq$ represents the temporal succession of worlds. We assume that there is a least world w_0 in the ordering, called the *initial world.*

The function a is a type-consistent assignment of values to predicates, functions and constants. We indicate with $a(w)(\alpha)$ the value of the symbol α at world $w \in W$. The assignment to rigid symbols does not depend on the world w.

We indicate with $\mathcal{I}, \mathcal{M} \models_w \phi$ the fact that the formula ϕ is true at world w of model $\mathcal{M}$ with variable assignment $\mathcal{I}$. Truth is computed by induction on the structure of ϕ in the usual way; as an example, the cases for $\Box$ and $\forall$ are:

$$\begin{array}{lll}\mathcal{I}, \mathcal{M} \models_w \Box\phi & \text{iff} & \forall w' \in W : w \leq w' \rightarrow \mathcal{I}, \mathcal{M} \models_{w'} \phi, \\ \mathcal{I}, \mathcal{M} \models_w \forall\xi\,\phi & \text{iff} & \forall d \in D_\xi : \mathcal{I}[d/\xi], \mathcal{M} \models_w \phi,\end{array}$$

where D_ξ is the domain corresponding to the type of ξ, and $\mathcal{I}[d/\xi]$ is the variable assignment obtained from $\mathcal{I}$ by assigning the value d to ξ.

Temporal logic and traces. We can use temporal logic to specify properties of traces by associating a model to each trace. We assume that functions and predicates have some predefined assignment.

To the discrete trace σ_d: $\langle s_0, t_0\rangle$, $\langle s_1, t_1\rangle$, $\langle s_2, t_2\rangle$, ... we associate the model $\mathcal{M}_{\sigma_d} = \langle \mathbb{N}, \leq, a_{\sigma_d}\rangle$, where a_{σ_d} is the assignment defined by, for $x \in \mathcal{V}$ and $n \in \mathbb{N}$:

$$a(n)(x) = s_n(x), \qquad a(n)(T) = t_n.$$

Instead of $\mathcal{I}, \mathcal{M}_{\sigma_d} \models_n \phi$, we will usually write $\mathcal{I}, \sigma_d \models_n \phi$.

In the model $\mathcal{M}_{\sigma_c}$ corresponding to a continuous trace σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle, \ldots$, we take as frame $\langle W, \leq\rangle$ the set of moments of σ_c together with their linear ordering; the initial world is $(0, 0)$. The assignment is then defined, for $x \in \mathcal{V}$ and $n \in \mathbb{N}$, by

$$a(n,t)(x) = r_n(x), \qquad a(n,t)(t) = t.$$

Again, we usually write $\mathcal{I}, \sigma_c \models_{(n,t)} \phi$ instead of $\mathcal{I}, \mathcal{M}_{\sigma_c} \models_{(n,t)} \phi$.

We can thus define two temporal logics: TL_D over discrete traces, and TL_C over continuous ones. A formula ϕ is valid in TL_D, written $\overset{\text{D}}{\models} \phi$, if $\mathcal{I}, \sigma_d \models_n \phi$ for all $\mathcal{I}$, σ_d, n. Similarly, ϕ is valid in TL_C, written $\overset{\text{C}}{\models} \phi$, if $\mathcal{I}, \sigma_c \models_{(n,t)} \phi$ for all $\mathcal{I}$, all σ_c, and all moments (n, t) of σ_c. In general, if one or more of the symbols $\mathcal{I}$, σ, w are omitted from $\mathcal{I}, \sigma \models_w \phi$, the truth of ϕ is required for all possible values of the omitted symbols.

Thus, $\models \phi$ means that ϕ is true in all the worlds of all the models. This semantics is called *floating semantics*, and is different from the *anchored* semantics presented in [19], in which $\models \phi$ means that ϕ is true in the *first* world of all models. This semantics has been chosen as it has simpler proof-theoretical properties, in the absence of a next-time operator.

We can also define the validity of formulas with respect to a system S by restricting the set of traces considered in the above definitions to those admitted by S. Correspondingly, we have the notions of a formula ϕ being S-valid in TL_D or TL_C, indicated respectively with $S \overset{\text{D}}{\models} \phi$, $S \overset{\text{C}}{\models} \phi$.

3.1 Specification and Verification

The logics TL_D and TL_C have different properties, reflecting the difference in the two underlying semantics.

Example 1 (density of time) The two logics TL_D, TL_C have different sets of valid formulas. For example, the formula

$$\phi:\ \forall\xi\,\forall\zeta\,\left[\Diamond(T = \xi) \wedge \Diamond(T = \zeta) \rightarrow \Diamond\left(T = \frac{\xi + \zeta}{2}\right)\right]$$

expressing the density of time is such that $\overset{\text{C}}{\models} \phi$, $\overset{\text{D}}{\models} \neg\phi$. ∎

While the continuous semantics corresponds closely to the physical behavior of the system, the discrete semantics gives only an approximate description in terms of a series of snapshots. System specifications, being ultimately a specification of the physical behavior, can be more faithfully expressed in the continuous semantics. For hybrid systems this is even truer, as the state can change continuously in time and continuous changes are not represented in the discrete semantics [18].

However, the verification of the properties of a system is simpler in the discrete semantics. The methods proposed in [7, 18, 20] to verify properties written in TL_D rely on two concepts: *verification conditions* and *verification rules.* If ϕ and ψ are arbitrary *past formulas*, that is, formulas not containing future temporal operators, it is possible to define the *verification conditions* $\{\phi\}\ \tau\ \{\psi\}$, $\{\phi\}$ **tick** $\{\psi\}$ having the following intuitive readings.

$\{\phi\}\ \tau\ \{\psi\}$: if ϕ is true, and the transition τ can be taken, ψ will be true in the resulting state.

$\{\phi\}$ **tick** $\{\psi\}$: if ϕ is true, and the time advances, ψ will be true in the resulting state.

The verification conditions allow in turn the statement of *verification rules* that relate the structure of the system to its temporal properties. An example of verification rule is the ubiquitous *invariance* rule:

$$\frac{S \overset{\text{D}}{\vdash} \Big\{\{\phi\}\ \tau\ \{\phi\}\Big\}_{\tau \in \mathcal{T}} \qquad S \overset{\text{D}}{\vdash} \{\phi\}\,\textbf{tick}\,\{\phi\}}{S \overset{\text{D}}{\vdash} \phi \rightarrow \Box\phi}.$$

The proof of the soundness of the verification conditions and of the verification rules makes an essential use of the discreteness of the semantics, so that the approach cannot be easily transferred to the continuous semantics.

3.2 Verification in the Continuous Semantics

In this paper we will show how the advantages of the discrete semantics can be transferred to the continuous one. The key idea consists in defining a property, *finite variability*, or FV, and showing that if ϕ is FV, then $S \overset{\text{D}}{\models} \phi$ implies $S \overset{\text{C}}{\models} \phi$.

To verify that a system satisfies a specification written in TL_C, we therefore propose a methodology consisting of three main ingredients.

The first one consists in the use of verification rules for TL_D, whose conclusion can be transferred to TL_C. This will enable us to go from the description of the structure of the system in terms of transitions to the properties it satisfies, expressed in temporal logic.

The second one is a series of axioms about time. These axioms state properties that are at the same time fundamental and not derivable in TL_D.

The third ingredient is a deductive system for TL_C. This will enable us to bring together the results of the verification in TL_D and of the axioms about time, leading to the desired real-time properties of a system. If it is desired, temporal reasoning in TL_C can often be kept to a minimum.

A related approach to proving $S \overset{\text{C}}{\models} \phi$ has been proposed in [8] for similar semantics and logics. It consists in *rephrasing* the property ϕ into a form ϕ' better suited to the discrete semantics. If the rephrasing is perfect, then $S \overset{\text{D}}{\models} \phi' \leftrightarrow S \overset{\text{C}}{\models} \phi$; otherwise, it is sometimes possible to find a stronger property ϕ' such that $S \overset{\text{D}}{\models} \phi' \rightarrow S \overset{\text{C}}{\models} \phi$. In [8] it is explained how to rephrase some formulas, and how to approximate others with stronger conditions.

Our approach extends the one based on rephrasing by considering general formulas. Moreover, since temporal reasoning in TL_C is allowed, we can prove the validity of formulas that have no useful rephrasing. Our strategy applies also to hybrid systems, where not only time but also other parameters of the state of the system can vary in a continuous way.

To show the soundness of our approach, we need a careful analysis of the relationship between the discrete and continuous semantics, to which we will now turn our attention.

4 From Discrete to Continuous Reasoning

4.1 Refinement

Each behavior of the system can be represented in more than one way by discrete or continuous traces, corresponding to the different ways of sampling the state of the system in time.

Example 2 The two discrete traces

$$\sigma_d : \overbrace{\langle x{=}0, t{=}0\rangle}^{0}, \overbrace{\langle x{=}1, t{=}0\rangle}^{1}, \qquad\qquad \overbrace{\langle x{=}1, t{=}10\rangle}^{2}, \cdots$$

$$\sigma_d' : \underbrace{\langle x{=}0, t{=}0\rangle}_{0}, \underbrace{\langle x{=}1, t{=}0\rangle}_{1}, \underbrace{\langle x{=}1, t{=}5\rangle}_{2}, \underbrace{\langle x{=}1, t{=}10\rangle}_{3}, \cdots$$

intuitively represent the same behavior of the system, but σ_d' contains one more sampling of the state of the system, $\langle x{=}1, t{=}5\rangle$. ∎

Specifically, we say that a trace is a *refinement* of another if it has been obtained by sampling the state of the system more frequently in time [15, 16, 2]. To give a formal definition of refinement, we introduce partitioning functions, that are closely related to the *event-stretching functions* of [13, 12].

Definition 6 (partitioning function) *A partitioning function μ is a function $\mathbb{N} \mapsto \mathrm{Int}_{\mathbb{N}}$ such that the intervals $\mu_0, \mu_1, \mu_2 \ldots$ are adjacent and disjoint. Formally, $\bigcup_{n\in\mathbb{N}} \mu_n = \mathbb{N}$, and $\forall n \in \mathbb{N} : \max \mu_i = \min \mu_{i+1} - 1$.*

Intuitively, a trace σ_d': $\langle s_0', t_0'\rangle$, $\langle s_1', t_1'\rangle$, $\langle s_2', t_2'\rangle$, $\ldots$ is a refinement of σ_d: $\langle s_0, t_0\rangle$, $\langle s_1, t_1\rangle$, $\langle s_2, t_2\rangle$, $\ldots$ if many observations of σ_d' correspond to a single observation of σ_d. We use the partitioning function to specify the correspondence: all the pairs $\langle s_j', t_j'\rangle$ with $j \in \mu_i$ will correspond to $\langle s_i, t_i\rangle$. Similarly, if σ_c: $\langle r_0', I_0'\rangle$, $\langle r_1', I_1'\rangle$, $\langle r_2', I_2'\rangle$, $\ldots$ is a refinement of σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle$, $\ldots$, all the intervals I_j with $j \in \mu_i$ will correspond to the single interval I_i.

Definition 7 (refinement) *A discrete trace σ_d': $\langle s_0', t_0'\rangle$, $\langle s_1', t_1'\rangle$, $\langle s_2', t_2'\rangle$, $\ldots$ is a refinement of σ_d: $\langle s_0, t_0\rangle$, $\langle s_1, t_1\rangle$, $\langle s_2, t_2\rangle$, $\ldots$ by the partitioning function μ, indicated by $\sigma_d' \succeq^{\mu} \sigma_d$, if for all i: $t'_{\min \mu_i} = t_i$, and $\forall j \in \mu_i : s_j' = s_i$.*

A continuous trace σ_c: $\langle r_0', I_0'\rangle$, $\langle r_1', I_1'\rangle$, $\langle r_2', I_2'\rangle$, $\ldots$ is a refinement of σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle$, $\ldots$ by the partitioning function μ, denoted $\sigma_c' \succeq^{\mu} \sigma_c$, if for all $i \in \mathbb{N}$:

$$I_i = \bigcup_{j\in\mu_i} I_j', \qquad \forall j \left(j \in \mu_i \rightarrow r_j' = r_i\right).$$

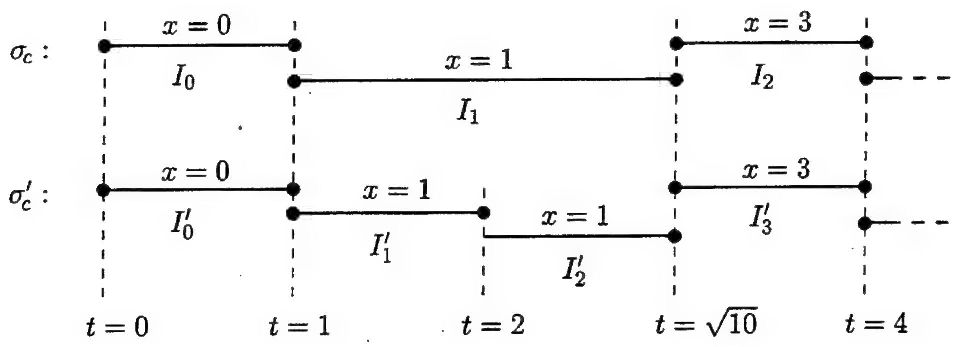


Figure 1: A closed continuous trace σ_c and one of its open refinements σ'_c: $\sigma'_c \succeq \sigma_c$.

Example 3 For σ_d, σ'_d as in Example 2, we have $\sigma'_d \succeq^\mu \sigma_d$ with $\mu_0 = \{0\}$, $\mu_1 = \{1,2\}$, $\mu_2 = \{3\}, \ldots$. Figure 1 gives an example of refinement of continuous computations. ∎

Note that the definition for continuous traces is independent of the fact that the trace is closed or not. In the following, we write σ to denote a generic trace, either discrete or continuous. We call *sample equivalent* two traces that have a common refinement [12].

Definition 8 (sample equivalence) *Two discrete (resp. continuous) traces σ, σ' are sample equivalent, written $\sigma \approx \sigma'$, if there is a discrete (resp. continuous) trace σ'' such that $\sigma'' \succeq \sigma$, $\sigma'' \succeq \sigma'$.*

Two sample equivalent traces are two different representations of the same behavior of the system. It is no surprise then that we have the following theorem, stating that systems do not distinguish between sample equivalent traces [15, 16].

Theorem 1 *If $\sigma_d \approx \sigma'_d$, then $S \triangleright \sigma_d$ iff $S \triangleright \sigma'_d$. If σ_c and σ'_c are both closed, and $\sigma_c \approx \sigma'_c$, then $S \triangleright \sigma_c$ iff $S \triangleright \sigma'_c$.*

In fact, it could be argued that a better representation of the behavior of the system can be obtained by considering equivalence classes of admitted traces modulo sampling equivalence. This equivalence classes, called *sample equivalence classes*, would be similar to the closure under stuttering of [2]. This is generally not done, as reasoning about equivalence classes of traces can be harder than reasoning about a single trace at a time.

Since sample equivalent traces correspond to the same behavior of the system, it is desirable that temporal logic does not distinguish among them. We say that a temporal logic is *sample invariant* if $\sigma \approx \sigma'$ implies $\mathcal{I}, \sigma \models \phi \leftrightarrow \mathcal{I}, \sigma' \models \phi$ [15]. The logic $\mathrm{TL_C}$ is sample invariant, $\mathrm{TL_D}$ is not. The result for $\mathrm{TL_C}$ is given by the following theorem, that establishes that if a trace is a refinement of another, the same formulas hold at corresponding moments.

Theorem 2 (sample invariance of $\mathrm{TL_C}$) *If $\sigma'_c \succeq^\mu \sigma_c$ and $j \in \mu_i$, then*

$$\mathcal{I}, \sigma'_c \models_{(j,t)} \phi \leftrightarrow \mathcal{I}, \sigma_c \models_{(i,t)} \phi.$$

If $\sigma'_c \approx \sigma_c$, then $\sigma'_c \models \phi \leftrightarrow \sigma_c \models \phi$.

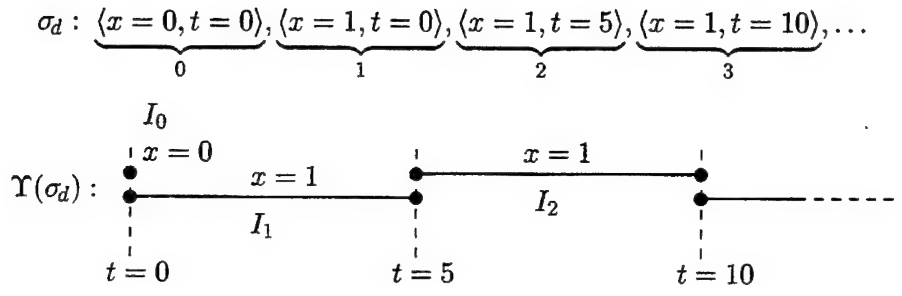

σ_d : $\underbrace{\langle x=0,t=0\rangle}_{0}, \underbrace{\langle x=1,t=0\rangle}_{1}, \underbrace{\langle x=1,t=5\rangle}_{2}, \underbrace{\langle x=1,t=10\rangle}_{3}, \ldots$
I_0
$x=0$
$\Upsilon(\sigma_d)$:
$x=1$
I_1
$x=1$
I_2
$t=0$
$t=5$
$t=10$

Figure 2: A discrete trace σ_d and its continuous translation $\Upsilon(\sigma_c)$.

4.2 Translations between Discrete and Continuous Semantics

To set up a correspondence between discrete and continuous traces that represent the same behavior of the system, we will use two translation functions: from discrete traces to continuous ones, and vice versa. These translations are uniquely determined between sample equivalence classes of traces, but we have some freedom to choose the trace that corresponds to a given one within a sample equivalence class.

The translation Υ from discrete traces to closed continuous traces associates to each $\langle s_n, t_n\rangle$ a closed interval stretching from t_n to t_{n+1}.

Definition 9 ($\Upsilon : \sigma_d \mapsto \sigma_c$) *We define the translation function Υ from discrete traces to continuous ones as the function associating to σ_d: $\langle s_0, t_0\rangle$, $\langle s_1, t_1\rangle$, $\langle s_2, t_2\rangle$, ... the closed trace σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle$, ... defined by, for all $n \in \mathbb{N}$: $r_n = s_n$, $I_n^{\leftarrow} = t_n$, $I_n^{\rightarrow} = t_{n+1}$.*

In the opposite translation, Ω, the idea is that each interval of the continuous trace is represented in the discrete trace by two observations, one for each endpoint. We define the translation so that also nonclosed traces can be translated, and some care must be taken to handle the case of open and half-open intervals.

Definition 10 ($\Omega : \sigma_c \mapsto \sigma_d$) *The translation function Ω associates to σ_c: $\langle r_0, I_0\rangle$, $\langle r_1, I_1\rangle$, $\langle r_2, I_2\rangle$, ... the discrete trace σ_d: $\langle s_0, t_0\rangle$, $\langle s_1, t_1\rangle$, $\langle s_2, t_2\rangle$, ... defined in the following way, for all $n \in \mathbb{N}$.*

1. $s_{2n} = s_{2n+1} = r_n$.
2. (a) *If I_n is closed, $t_{2n} = I_n^{\leftarrow}$, $t_{2n+1} = I_n^{\rightarrow}$.*
   (b) *If I_n is left open, $t_{2n} = t_{2n+1} = I_n^{\rightarrow}$.*
   (c) *If I_n is right open, $t_{2n} = t_{2n+1} = I_n^{\leftarrow}$.*
   (d) *If I_n is open, $t_{2n} = t_{2n+1} = (I_n^{\leftarrow} + I_n^{\rightarrow})/2$.*

Figures 2 and 3 show examples of traces and their translations. The following lemma shows that the translations are one the inverse of the other, modulo sampling equivalence,

$$\Omega(\sigma_c):\ \underbrace{\langle x{=}0, t{=}0\rangle}_{0}, \underbrace{\langle x{=}0, t{=}1\rangle}_{1}, \underbrace{\langle x{=}1, t{=}4\rangle}_{2}, \underbrace{\langle x{=}1, t{=}4\rangle}_{3}, \underbrace{\langle x{=}7, t{=}5\rangle}_{4}, \underbrace{\langle x{=}7, t{=}5\rangle}_{5}, \underbrace{\langle x{=}2, t{=}6\rangle}_{6}, \ldots$$

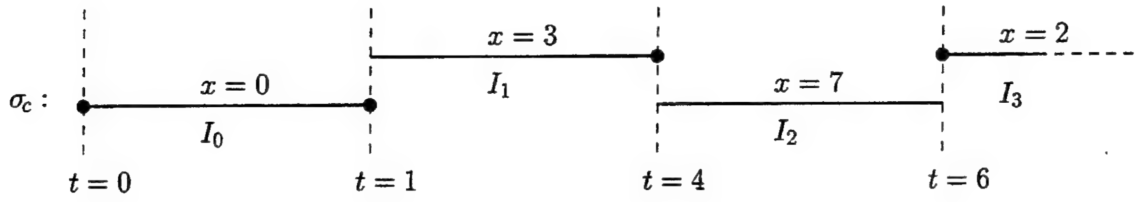


Figure 3: An open trace σ_c and its discrete translation $\Omega(\sigma_c)$. Note that σ_c is not the refinement of any closed trace.

and that they preserve for closed traces the partial order of refinement of traces. It also suggests that traces related by the translation functions represent the same behavior of the system.

Lemma 1

1. *For any σ_d, σ_d', σ_c, σ_c', with σ_c closed, we have:*

$$\sigma_d' \succeq \sigma_d \to \Upsilon(\sigma_d') \succeq \Upsilon(\sigma_d), \qquad \Omega(\Upsilon(\sigma_d)) \succeq \sigma_d,$$
$$\sigma_c' \succeq \sigma_c \to \Omega(\sigma_c') \succeq \Omega(\sigma_c), \qquad \Upsilon(\Omega(\sigma_c)) \succeq \sigma_c.$$

2. *For any S, σ_d and closed σ_c, $S \rhd \sigma_d$ iff $S \rhd \Upsilon(\sigma_d)$, and $S \rhd \sigma_c$ iff $S \rhd \Omega(\sigma_c)$.*

3. *If $S \rhd \sigma_c$ and $\sigma_c' \succeq \sigma_c$, then $S \rhd \Omega(\sigma_c')$.*

4.3 Finite Variability

Consider the formula $T > 3 \vee x = 4$. In every finite interval of a continuous trace, the truth value of its subformulas can change at most a finite number of times. Thus, given a trace σ_c, it seems possible to refine it into a (possibly open) trace σ_c: $\langle r_0', I_0'\rangle$, $\langle r_1', I_1'\rangle$, $\langle r_2', I_2'\rangle$, ... such that each subformula has constant truth value throughout all intervals I_j', $j \in \mathbb{N}$. This is the idea underlying the definition of finite variability.

The set of subformulas of ϕ, denoted by $\mathrm{sb}(\phi)$, is defined by induction on the structure of ϕ:

$$\mathrm{sb}(P\,u_1 \ldots u_n) = \{P\,u_1 \ldots u_n\} \cup \textstyle\bigcup_{i=1}^{n} \mathrm{sb}(u_i)$$
$$\mathrm{sb}(u_1 = u_2) = \{u_1 = u_2\} \cup \mathrm{sb}(u_1) \cup \mathrm{sb}(u_2)$$
$$\mathrm{sb}(\neg\phi) = \{\neg\phi\} \cup \mathrm{sb}(\phi)$$
$$\mathrm{sb}(\phi \to \psi) = \{\phi \to \psi\} \cup \mathrm{sb}(\phi) \cup \mathrm{sb}(\psi)$$
$$\mathrm{sb}(\Box\phi) = \{\Box\phi\} \cup \mathrm{sb}(\phi)$$

$$\mathrm{sb}(\phi\,\mathcal{U}\,\psi) = \{\phi\,\mathcal{U}\,\psi\} \cup \mathrm{sb}(\phi) \cup \mathrm{sb}(\psi)$$
$$\mathrm{sb}(\forall x\,\phi) = \{\forall x\,\phi\} \cup \mathrm{sb}(\phi)$$

and similarly for the other propositional connectives and temporal operators. The set of subformulas of a term is defined by:

$$\mathrm{sb}(c) = \emptyset \qquad\qquad \mathrm{sb}(\xi) = \emptyset$$
$$\mathrm{sb}(f\,u_1 \dots u_n) = \textstyle\bigcup_{i=1}^{n} \mathrm{sb}(u_i) \qquad \mathrm{sb}(\Gamma(\phi)) = \mathrm{sb}(\phi),$$

where c denotes a constant, flexible or rigid. Finite variability can then be defined as follows.

Definition 11 (finite variability) *A formula ϕ has the property of finite variability, or FV, if for every closed σ_c and every $\mathcal{I}$ there exists a $\sigma_c' \succeq \sigma_c$ such that*

$$\mathcal{I}, \sigma_c' \models_{(i,t)} \psi \leftrightarrow \mathcal{I}, \sigma_c' \models_{(i,t')} \psi$$

for all subformulas $\psi \in \mathrm{sb}(\phi)$. The trace σ_c' with the above property can be open, and is called a ground trace for ϕ, σ_c and $\mathcal{I}$.

Example 4 Many common formulas used in the specification and verification of systems are FV. On the other hand, an example of a formula which is not FV is the following:

$$T < 4 \rightarrow \Diamond \left[\cos \frac{1}{T-4} > 0\right].$$

The reason why the above formula is not FV is that it is not possible to subdivide $\mathbb{R}^+$ into a finite number of intervals in which the subformula $\cos(1/(T-4)) > 0$ has constant value. ∎

Example 5 Another, more subtle, example of a formula which is not FV is given by the formula ϕ of Example 1. The reason why it is not possible to refine a given σ_c into a σ_c' such that the values of the subformulas are constant in the intervals of σ_c' has to do with the way quantification interacts with time. Specifically, for each value of ξ and ζ it is possible to find a σ_c' such that the subformulas $\xi = T$, $\zeta = T$ and $T = (\xi + \zeta)/2$ have constant value in the intervals. However, it is not possible to find a σ_c' that has this property for all possible values of ξ and ζ. ∎

The importance of the concept of finite variability lies in the fact that if all subformulas have constant truth value throughout an interval, then the ground continuous trace is faithfully represented by its discrete translation. The necessity of considering formulas that have constant truth value in the intervals had already been recognized in [20], where the set of *important events* was introduced purposely to prevent certain formulas from changing truth value in an interval. The definition of finite variability provides a more general solution: it gives an account of the behavior of quantification, and it allows to change the temporal logic specifications without also having to change the set of important events.

For FV formulas, the connection between $\mathrm{TL_C}$ and $\mathrm{TL_D}$ is expressed by the following results.

Theorem 3 *If σ_c' is a ground trace for ϕ, σ_c, $\mathcal{I}$, with σ_c closed, then*

$$\mathcal{I}, \Omega(\sigma_c') \models_{2n} \phi \leftrightarrow \mathcal{I}, \sigma_c' \models_{(n,t)} \phi.$$

This theorem enables us to make a connection between the formulas that are valid, or S-valid, in the two logics.

Theorem 4 (transfer of validity) *If $S \overset{\mathrm{D}}{\models} \phi$ and ϕ is FV, then $S \overset{\mathrm{C}}{\models} \phi$. If $\overset{\mathrm{D}}{\models} \phi$ and ϕ is FV, then $\overset{\mathrm{C}}{\models} \phi$.*

Proof. We prove only the first statement, as the proof of the second is similar. We prove the counterpositive: assume $S \overset{\mathrm{C}}{\not\models} \phi$. Then there are $\mathcal{I}$, σ_c and a moment (n,t) of σ_c such that $S \triangleright \sigma_c, \mathcal{I}, \sigma_c \not\models_{(n,t)} \phi$. As ϕ is FV and σ_c is closed, there is a trace $\sigma_c' \succeq^{\mu} \sigma_c$ that is ground for ϕ, σ_c, $\mathcal{I}$. There is a $k \in \mu_n$ such that (k,t) is a moment of σ_c', and from Theorem 2 we have that $\mathcal{I}, \sigma_c' \not\models_{(k,t)} \phi$. As σ_c' is ground for $\mathcal{I}$, ϕ, by Theorem 3 we have $\mathcal{I}, \Omega(\sigma_c') \not\models_{2k} \phi$. Lemma 1 ensures that $S \triangleright \Omega(\sigma_c')$, and we finally get $S \overset{\mathrm{D}}{\not\models} \phi$, which concludes the proof. ∎

Note that the converse of this theorem does not hold, i.e. if ϕ is FV and $S \overset{\mathrm{C}}{\models} \phi$, it does not follow that $S \overset{\mathrm{D}}{\models} \phi$. A simple example is provided by $\phi : \ \Diamond(T = 5)$, which is valid in the continuous semantics, but is not necessarily valid on a discrete trace of a system (see Example 2).

4.4 From Discrete to Continuous Validity

Finite variability is a semantic property of a formula: to be able to use the result of the last theorem in a proof system for TL$_\mathrm{C}$, we need to replace it by some syntactic criterion.

To obtain a sufficient syntactical condition for FV, we first define *well-behaved* functions that are analytical along the real axis in some of their variables. Here, the word "analytical" is used in the calculus sense.

Definition 12 (well-behaved function) *We say that a function $f(z_0, \ldots, z_n, v_1, \ldots, v_k)$ is well-behaved if, for all $1 \leq i \leq n$, and for all real $z_{j \neq i}$, v_m $(1 \leq j \leq n,\ 1 \leq m \leq k)$, f when considered as a function of z_i only is analytical in a region of the complex plane containing the real axis.*

Example 6 Examples of well-behaved functions are

$$\begin{aligned} f(z_0, z_1, v_0) &= z_0 + z_1 + v_0, \\ f(z_0, v_0) &= |v_0| + z_0, \\ f(z_0) &= 1/(2 + z_0^2), \\ f(z_0, z_1, v_0, v_1) &= \sin(v_0 z_0)\cos(v_1 z_1). \end{aligned}$$

The function $f(z_0) = z_0^3 \sin(1/z_0)$, on the other hand, is not well-behaved, as when considered as a function of z_0 it is not analytical in $z_0 = 0$. ∎

Definition 13 (syntactic finite variability (SFV)) *We call SFV the formulas that are constructed in the following inductive way.*

1. *If $u_1, \ldots, u_n$ are terms not containing T or Γ, then $Pu_1 \ldots u_n$ is SFV.*

2. *If $f(z_0, \ldots, z_n, v_1, \ldots, v_k)$ is a well-behaved function, then*

$$f\Big(T, \Gamma(\phi_1), \ldots, \Gamma(\phi_n), c_1, \ldots, c_k\Big) = 0,$$
$$f\Big(T, \Gamma(\phi_1), \ldots, \Gamma(\phi_n), c_1, \ldots, c_k\Big) > 0,$$

   where $c_1, \ldots, c_k$ are either constants different from T or variables, and $\phi_1, \ldots \phi_n$ do not contain T or Γ, is a SFV formula. We call this type of SFV formulas T-atoms.

3. *A formula constructed from SFV formulas using propositional connectives or temporal operators is a SFV formula.*

4. *If ϕ is a SFV formula, and ξ does not occur in any T-atom of ϕ, then $\forall \xi\, \phi$ is a SFV formula.*

Within an interval of a continuous trace σ_c, the $c_1, \ldots, c_k$ of the above definition have constant value. The requirement that $f(z_0, \ldots, z_n, v_1, \ldots, v_k)$ is well-behaved insures that within each interval of σ_c the inequalities change truth value at most finitely often. This is a consequence of a well-known theorem of calculus stating that a function can have at most a finite number of zeroes in a finite region of the complex plane where it is analytical.

We will say that a formula is SFV even if it is not in a form described by the above definition, but can be easily transformed and put in such a form. As an example, $T > x + y$ is not in the form defined above, but it can be transformed into $T - x - y > 0$, and will thus also be called SFV. In a similar way, $T \geq \Gamma(x = 2) + 4$ can be transformed in $[T - \Gamma(x = 2) - 4 = 0] \vee [T - \Gamma(x = 2) - 4 > 0]$ which is of the above form. It is possible to give a more general definition of SFV that encompasses directly all these cases, but it would be far less concise.

Example 7 The formula ϕ of Example 4 is not SFV, as the function $\cos(1/(x - 4))$ is not analytical in $x = 4$, a point of the real axis. The formula of Example 1 is not SFV as it quantifies over ξ and ζ that appear in the T-atoms $T = \xi$, $T = \zeta$ and $T = (\xi + \zeta)/2$. ∎

We have that SFV implies FV, as the theorem below states.

Theorem 5 (SFV implies FV) *If ϕ is SVF, it is also FV.*

Corollary 1 *If ϕ is SFV, $S \overset{\text{D}}{\models} \phi$ implies $S \overset{\text{C}}{\models} \phi$. Similarly for initial validity. Therefore the inference rules*

$$\frac{S \overset{\text{D}}{\vdash} \phi}{S \overset{\text{C}}{\vdash} \phi}, \qquad \frac{\overset{\text{D}}{\vdash} \phi}{\overset{\text{C}}{\vdash} \phi}, \qquad \frac{S \overset{\text{D}}{\vdash}_0 \phi}{S \overset{\text{C}}{\vdash}_{(0,0)} \phi}, \qquad \frac{\overset{\text{D}}{\vdash}_0 \phi}{\overset{\text{C}}{\vdash}_{(0,0)} \phi},$$

with the proviso that ϕ is SFV, are sound.

Using syntactic finite variability, we can also establish a connection with propositional temporal logic. Let PTL be the propositional temporal logic of discrete linear time, on the frame $\langle \mathbb{N}, \leq \rangle$, with temporal operators $\mathcal{U}$, $\mathcal{S}$, $\Box$, $\boxminus$, $\Diamond$, $\diamondminus$, and based on the floating semantics. This logic is the same as the one presented in [17], apart for the absence of $\bigcirc$, $\ominus$. The following results hold.

Theorem 6 (from PTL to TL$_\mathrm{C}$) *If $\models^{\mathrm{PTL}} \alpha[p_1, \ldots p_n]$, where $p_1, \ldots, p_n$ are propositional letters, then $\models^{\mathrm{C}} \alpha[\phi_1, \ldots, \phi_n]$ provided $\phi_1, \ldots, \phi_n$ are FV. Similarly for initial validity. Therefore, the following inference rules*

$$\frac{\vdash^{\mathrm{PTL}} \alpha[p_1, \ldots p_n]}{\vdash^{\mathrm{C}} \alpha[\phi_1, \ldots, \phi_n]}, \qquad \frac{\vdash_0^{\mathrm{PTL}} \alpha[p_1, \ldots p_n]}{\vdash_{(0,0)}^{\mathrm{C}} \alpha[\phi_1, \ldots, \phi_n]},$$

with the proviso that $\phi_1, \ldots, \phi_n$ are SFV, are sound.

It is well known that a similar result holds for TL$_\mathrm{D}$, for which FV is not required [19]. This result is of relevant practical importance, because deductive systems for PTL are well-studied [17], and efficient decision algorithms for the problem of initial validity exist [11].

4.5 Reasoning in the Continuous Semantics

Sometimes it is necessary to carry out a small part of the reasoning in the continuous semantics, to put together the results of the verification rules and reach the desired conclusion. In practical verification examples, most of this reasoning is limited to using simple axioms about the completeness and divergence of time along any continuous trace. It is possible to give an axiomatization for TL$_\mathrm{C}$. As temporal logic with past, future and explicit time is incomplete [4, 1, 2], this axiomatization will also be incomplete for the first-order case, but nonetheless it will allow the proof of many formulas that arise in practice. The axioms can be divided in three categories: propositional, first-order and about time.

Propositional axioms. The frame $\langle W, \leq \rangle$ of a model $\mathcal{M}_{\sigma_c}$ derived from a trace σ_c is neither discrete, nor dense, nor complete. In fact, in each interval the set of moments is complete, but there is no moment between the two endpoints of two adjacent closed intervals. We will therefore use an axiomatization for the general frame $\langle W, \leq \rangle$ with the only hypothesis that it is a reflexive linear order with initial world.

Unfortunately, there is no complete set of axioms available in the literature for temporal logic with $\mathcal{U}$, $\mathcal{S}$ and the other temporal operators over the frame $\langle W, \leq \rangle$. A complete axiomatization for $\mathcal{U}$ and $\mathcal{S}$ over the frame $\langle W, < \rangle$ has been presented in [3], and it is possible to adapt those axioms schemas to a reflexive frame, but no claim of completeness is made at this point. The adapted axioms schemas are listed in Table 1. Of all these axioms, except the one marked with (‡), also the *specular image* should be taken as an axiom [21]. The specular image of a temporal formula is the formula obtained by substituting the future operators with the corresponding past operators, and vice versa. For example, the specular image of $\Box \diamondminus (T = 5)$ is $\boxminus \Diamond (T = 5)$.

All propositional logic tautologies.	$\phi\,\mathcal{U}\,\psi \to \Diamond\psi$
$\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi)$	$\Box(\phi \to \psi) \to (\phi\,\mathcal{U}\,\gamma) \to (\psi\,\mathcal{U}\,\gamma)$
$\phi \to \Box\ominus\phi$	$\Box(\phi \to \psi) \to (\gamma\,\mathcal{U}\,\phi) \to (\gamma\,\mathcal{U}\,\psi)$
$\Box\phi \to \Box\Box\phi$	$\phi\,\mathcal{U}\,\psi \leftrightarrow (\phi\,\mathcal{U}\,\psi)\,\mathcal{U}\,\psi$
$\Box\phi \wedge \boxminus\phi \to \Box\boxminus\phi$	$\phi\,\mathcal{U}\,\psi \leftrightarrow \phi\,\mathcal{U}\,(\phi\,\mathcal{U}\,\psi)$
$\Box\phi \to \phi$	$\phi\,\mathcal{U}\,\psi \wedge \neg(\gamma\,\mathcal{U}\,\psi) \to \phi\,\mathcal{U}\,(\phi \wedge \neg\psi)$
$\Box\phi \leftrightarrow \neg\Diamond\neg\phi$	$\phi\,\mathcal{U}\,\psi \to \phi \vee \psi$
$\ominus(\boxminus\phi \vee \boxminus\neg\phi) \quad (\ddagger)$	$\Diamond\phi \to (\psi \vee \neg\psi)\,\mathcal{U}\,\phi$
$\phi\,\mathcal{U}\,\psi \wedge \gamma\,\mathcal{U}\,\delta \to (\phi \wedge \gamma)\,\mathcal{U}\left[\left(\psi \wedge (\gamma\,\mathcal{U}\,\delta)\right) \vee \left(\delta \wedge (\phi\,\mathcal{U}\,\psi)\right)\right]$	
$\phi \wedge (\psi\,\mathcal{U}\,\gamma) \to \psi\,\mathcal{U}\left[\gamma \wedge (\psi \vee \gamma)\,\mathcal{S}\,\phi\right]$	

Table 1: Propositional axiom schemas for $\mathrm{TL_C}$.

$\pi = \pi$	$P\pi_1 \ldots \pi_n \to \Box P\pi_1 \ldots \pi_n \quad (\dagger\dagger)$
$\pi_1 = \pi_2 \to \phi\langle\pi_1\rangle \to \phi\langle\pi_2\rangle$	$\neg P\pi_1 \ldots \pi_n \to \Box\neg P\pi_1 \ldots \pi_n \quad (\dagger\dagger)$
$\pi_1 = \pi_2 \to \Box(\pi_1 = \pi_2) \quad (\dagger\dagger)$	$\forall x\,\Box\phi \to \Box\forall x\,\phi$
$\neg(\pi_1 = \pi_2) \to \Box\neg(\pi_1 = \pi_2) \quad (\dagger\dagger)$	

Table 2: First-order axiom schemas for $\mathrm{TL_C}$. The axioms denoted by ($\dagger\dagger$) have the proviso that $\pi_1, \pi_2, \ldots, \pi_n$ are terms not containing any flexible constant.

Another way of proceeding consists in defining the reflexive operators in terms of the irreflexive ones, that is, recursively rewrite each $p\,\mathcal{U}\,q$ in $p \wedge (p\,\mathcal{U}\,q)$, and similarly for $\mathcal{S}$ (the other operators can be defined from these two), and then use the original axiomatization proposed in [3] on the translation. Some additional axiom is still necessary to account for the presence of an initial world.

First-order axioms. The set of first-order axioms we will use is entirely classical. They account for rigid and flexible constants and equality, and they include the Barcan Formula, as the domains of quantification are rigid. A list of axiom schemas is given in Table 2. In the table, if $\phi\langle\pi_1\rangle$ is a formula containing the term π_1, $\phi\langle\pi_2\rangle$ denotes a formula obtained from $\phi\langle\pi_1\rangle$ by replacing some occurrences of π_1 with π_2, provided no free variable of π_2 is captured in the process.

Moreover, we the additional axiom $S \vdash^{C}_{(0,0)} \Theta_f$ states that all traces of a system start in an initial state.

$$
\begin{array}{ll}
\vdash^{C}_{(0,0)} T = 0 & T = \xi \rightarrow \Box(T \geq \xi) \\
T \geq 0 & \xi \geq T \rightarrow \Diamond(T = \xi) \\
\neg\phi \rightarrow \Gamma(\phi) = 0 & 0 \leq \Gamma(\phi) \leq T
\end{array}
$$

$$\Gamma(\phi) = \xi \wedge \zeta \geq \xi \rightarrow \Diamond(\phi \rightarrow \Gamma(\phi) = \zeta)$$

$$T = \xi + \zeta \wedge \phi\, \mathcal{S} \left(T = \xi \wedge \Gamma(\phi) = v\right) \rightarrow \Gamma(\phi) = v + \zeta$$

$$T = \xi \wedge \boxminus\neg\left[\phi\, \mathcal{U}\, (T = \xi)\right] \rightarrow \Gamma(\phi) = 0$$

Table 3: Time axiom schemas for TL_C.

$$
\begin{array}{cccc}
\dfrac{\vdash^{C} \phi \rightarrow \psi, \quad \vdash^{C} \phi}{\vdash^{C} \psi} &
\dfrac{\vdash^{C}_{(0,0)} \phi \rightarrow \psi, \quad \vdash^{C}_{(0,0)} \phi}{\vdash^{C} \psi} &
\dfrac{\vdash^{C} \phi}{\vdash^{C} \Box\phi} &
\dfrac{\vdash^{C} \phi}{\vdash^{C} \boxminus\phi} \\[2ex]
\dfrac{\vdash^{C} \phi \rightarrow \psi}{\vdash^{C} \phi \rightarrow \forall\xi\, \psi}(\dagger) &
\dfrac{\vdash^{C}_{(0,0)} \phi \rightarrow \psi}{\vdash^{C}_{(0,0)} \phi \rightarrow \forall\xi\, \psi}(\dagger) &
\dfrac{\vdash^{C}_{(0,0)} \Box\phi}{\vdash^{C} \Box\phi} &
\dfrac{\vdash^{C} \phi}{\vdash^{C}_{(0,0)} \phi} \\[2ex]
\dfrac{\vdash^{PTL} \alpha[p_1, \ldots p_n]}{\vdash^{C} \alpha[\phi_1, \ldots, \phi_n]}(\S) &
\dfrac{\vdash^{PTL}_{0} \alpha[p_1, \ldots p_n]}{\vdash^{C}_{(0,0)} \alpha[\phi_1, \ldots, \phi_n]}(\S) &
\dfrac{\vdash^{D} \phi}{\vdash^{C} \phi}(\S) &
\dfrac{\vdash^{D}_{0} \phi}{\vdash^{C}_{(0,0)} \phi}(\S)
\end{array}
$$

Table 4: Inference rules. The rules denoted by (†) have the proviso that ξ must not occur free in ϕ. The rules denoted by (§) have the proviso that ϕ, ϕ_1, ..., ϕ_n are SFV. In all of them, if the premiss(es) is (are) S-valid, the conclusion is S-valid.

Time axioms. A final set of axioms, listed in Table 3, are used to reason about time. As usual, we list an axiom ϕ to mean $\vdash^{C} \phi$: in the case where we claim only the initial validity of the axiom, as in the case of the first one, we write it explicitly.

Inference rules. The inference rules we propose are listed in Table 4. Note that these rules are based on the floating semantics. On the other hand, the verification rules that have been proposed in [6, 20] are based on the anchored semantics. To transfer the results from the anchored to the floating semantics, it suffices to use the rules:

$$\frac{\vdash^{Da} \Box\phi}{\vdash^{D} \phi}, \qquad \frac{S \vdash^{Da} \Box\phi}{S \vdash^{D} \phi},$$

where $\vdash^{Da}$ is the provability relation in the anchored version of TL_D.

5 An Example of Verification

We will now present a simple example of how the verification methods for $\mathrm{TL_D}$ can be used together with the time axioms and temporal reasoning to prove simple properties of systems expressed in $\mathrm{TL_C}$. We will choose a property that does not hold in $\mathrm{TL_D}$, to demonstrate the use of the time axioms for $\mathrm{TL_C}$.

We will not enter in the details of how the verification rules for $\mathrm{TL_D}$ are used to prove properties of a system, as this topic is dealt with in detail in [18, 9, 20].

Imprecise Oscillator

Consider a system OSC, consisting of an oscillator whose state is represented by the variable x. The oscillator can be in any of two states, $x = 0$ and $x = 1$, and it can stay in each of them for 3 to 5 seconds before switching to the other one. The oscillator start in the state $x = 0$. The system can be described by:

$$\begin{array}{rlrl} \Theta_f: & x=0 & \rho_{\tau_0}: & x=0 \wedge x'=1 \\ \mathcal{T}: & \{\tau_0,\tau_1\} & \rho_{\tau_1}: & x=1 \wedge x'=0 \\ l_{\tau_0}, l_{\tau_1}: & 3 & u_{\tau_0}, u_{\tau_1}: & 5 \end{array}$$

We want to verify that OSC satifies the following property:

> "The oscillator is in the state $x = 1$ some time between 6 and 7 seconds after it is started."

This specification can be written as

$$\text{OSC} \overset{\mathrm{C}}{\models}_{(0,0)} \Diamond(x = 1 \wedge 6 < T < 7). \tag{1}$$

It is not difficult to see that the corresponding specification in $\mathrm{TL_D}$, $\text{OSC} \overset{\mathrm{D}}{\models}_0 \Diamond(x = 1 \wedge 6 < T < 7)$, does not hold. To prove (1), define the abbreviations

$$\psi: \quad x = 0 \wedge \Gamma(x = 0) = T \wedge T \leq 3, \tag{2}$$

$$\phi: \quad T \leq 8 \rightarrow \Big[x = 1 \wedge \Gamma(x = 1) \leq T - 3\Big]. \tag{3}$$

The following implications hold:

$$\phi \rightarrow \Big(T = 6.5 \rightarrow x = 1\Big), \qquad \psi \rightarrow \Big(T = 6.5 \rightarrow x = 1\Big). \tag{4}$$

The proof of the specification (1) proceeds as follows.

$\text{OSC} \overset{\mathrm{D}}{\vdash}_0 \psi \mathcal{W} \phi$	from wait-for verification rule for $\mathrm{TL_D}$	(5)
$\text{OSC} \overset{\mathrm{D}}{\vdash} \phi \rightarrow \Box\phi$	from invariance verification fule for $\mathrm{TL_D}$	(6)
$\text{OSC} \overset{\mathrm{D}}{\vdash}_0 \psi \mathcal{W} \Box\phi$	from (5), (6) by temporal reasoning in $\mathrm{TL_D}$	(7)
$\text{OSC} \overset{\mathrm{D}}{\vdash} \Box(T = 6.5 \rightarrow x = 1)$	from (4), (5), (6) by temporal reasoning in $\mathrm{TL_D}$	(8)

$\mathrm{osc} \vdash^{\mathrm{C}}_{(0,0)} \Box(T = 6.5 \rightarrow x = 1)$ from (8), as it is SFV (9)

$\mathrm{osc} \vdash^{\mathrm{C}}_{(0,0)} \Diamond(T = 6.5)$ from the time axioms of $\mathrm{TL_C}$ (10)

$\mathrm{osc} \vdash^{\mathrm{C}}_{(0,0)} \Diamond(T = 6.5 \wedge x = 1)$ from (9), (10), temporal reasoning in $\mathrm{TL_C}$ (11)

$\mathrm{osc} \vdash^{\mathrm{C}}_{(0,0)} \Diamond(x = 1 \wedge 6 < T < 7)$ from (11) (12)

It is also possible to eliminate from this proof all temporal reasoning in $\mathrm{TL_C}$, apart from the application of the time axioms. This is done by introducing antecedents of implications in $\mathrm{TL_D}$ that will be discarded by time axioms of $\mathrm{TL_C}$. This transformation shows how reasoning in $\mathrm{TL_C}$ can be kept to a minimum. The final steps of the previous proof can be modified as follows.

$\mathrm{osc} \vdash^{\mathrm{D}}_{0} \Diamond(T = 6.5) \rightarrow \Diamond(T = 6.5 \wedge x = 1)$ from (8) by temp. reas. in $\mathrm{TL_D}$(13)

$\mathrm{osc} \vdash^{\mathrm{D}}_{0} \Diamond(T = 6.5) \rightarrow \Diamond(x = 1 \wedge 6 < T < 7)$ from (13) (14)

$\mathrm{osc} \vdash^{\mathrm{C}}_{(0,0)} \Diamond(T = 6.5) \rightarrow \Diamond(x = 1 \wedge 6 < T < 7)$ from (14), as it is SFV (15)

$\vdash^{\mathrm{C}}_{(0,0)} \Diamond(T = 6.5)$ from the time axioms of $\mathrm{TL_C}$ (16)

$\mathrm{osc} \vdash^{\mathrm{C}}_{(0,0)} \Diamond(x = 1 \wedge 6 < T < 7)$ from (15), (16) (17)

6 Hybrid Systems

The results obtained for real-time systems can be transferred to hybrid systems, provided that a proper relationship can be set up between the discrete and continuous semantics. In particular, we need to give a new definition of SFV for hybrid systems, to account for the fact that the state can change continuously in time, and we need to show how to define the traces and the translations in such a way that we can prove the analogous of Theorem 4.

6.1 Phase Transition Systems

We will model hybrid systems by *phase transition systems* similar those of [18, 20]. A phase transition system (PTS) $S = \langle \mathcal{V}, \Sigma, \mathcal{P}, \mathcal{T}, L, U, \Theta \rangle$ consists of the following components.

1. A set $\mathcal{V}$ of variables, called state variables, each with its type. $\mathcal{V}$ is partitioned into two disjoint subsets: $\mathcal{V}_d$ and $\mathcal{V}_c$. The variables in $\mathcal{V}_d$ are called *discrete* variables, they can be of any type and they can change only in an instantaneous way. The variables in $\mathcal{V}_c$ are called continuous variables, have type *real*, and can change both in an instantaneous and in a continuous way.

2. A set Σ of states: each state is a type consistent interpretation of the variables. Again, we write $s(x)$ to denote the interpretation of $x \in \mathcal{V}$ at state s. We write $s|_{\mathcal{V}_d}$, $s|_{\mathcal{V}_c}$ to denote the restrictions of the interpretation s to discrete and continuous variables only, respectively.

3. A set $\Theta \subseteq \Sigma$ of initial states.

4. A set $\mathcal{P}$ of *phases*. $\mathcal{P}$ is partitioned into disjoint subsets, one for each variable in $\mathcal{V}_c$. The subset corresponding to $x \in \mathcal{V}_c$ will be denoted by $\mathcal{P}_x$.
5. A set $\mathcal{T}$ of transitions, where $\tau \subseteq \Sigma \times \Sigma$ for each $\tau \in \Sigma$. $\mathcal{T}$ is partitioned into two disjoint subsets $\mathcal{T}_i$ and $\mathcal{T}_d$. The set $\mathcal{T}_i$ is the set of *immediate* transitions, that must be executed no later than the time at which they become enabled. The set $\mathcal{T}_d$ is the set of *delayed* transitions, whose enabling does not depend on the continuous variables.
6. Two sets L, U of minimum and maximum delays for the transitions in $\mathcal{T}_d$.

Phases. For each $x \in \mathcal{V}_c$, every phase $\wp \in \mathcal{P}_x$ is composed of an *enabling condition* $c_\wp \subseteq \Sigma$ and of a *phase function* $f_\wp : \Sigma \mapsto \mathbb{R}$. The phase $\wp$ is used to represent a differential equation governing x: the intended meaning is that if $c_\wp$ holds, then it must be $\dot{x} = f_\wp(s)$ in each state s where the state changes continuously. The enabling condition $c_\wp$ can depend on the discrete variables only: formally, for all $s, s' \in \Sigma$, $s|_{\mathcal{V}_d} = s'|_{\mathcal{V}_d} \rightarrow (s \in c_\wp \leftrightarrow s' \in c_\wp)$.

We say that a phase $\wp$ is *linear* if the function $f_\wp$ is a linear function of the continuous variables. It is not required that $f_\wp$ is linear in the discrete variables as well.

Transitions. We define the *enabling condition* c_τ of a transition $\tau \in \mathcal{T}$ as the set of states that have a successor according to the transition, or $c_\tau = \{s \mid \exists s'[(s, s') \in \tau]\}$. Transitions must be self-disabling, that is, $(s, s') \in \tau \rightarrow s' \notin c_\tau$.

If an immediate transition becomes enabled at time t, it has to be taken or disabled by some other transition before time advances past t. There is no restriction on the enabling condition of immediate transitions: it can depend on both the continuous and the discrete part of the state.

Each delayed transition $\tau \in \mathcal{T}_d$ has an associated minimum delay $l_\tau \in L$ and maximum delay $u_\tau \in U$, with $0 \leq l_\tau \leq u_\tau \leq \infty$. After τ is enabled, it can wait for a time t_d: $l_\tau \leq t_d \leq u_\tau$ before being taken. The enabling condition of delayed transitions can depend only on the discrete component of the state: for all $s, s' \in \Sigma$, it is $s|_{\mathcal{V}_d} = s'|_{\mathcal{V}_d} \rightarrow (s \in c_\wp \leftrightarrow s' \in c_\wp)$.

6.2 Continuous Semantics

The continuous semantics of hybrid systems is defined in terms of *hybrid traces*. They differ from the continuous traces used for real-time systems, as the value of the continuous variables can vary in the intervals composing the trace. The definition is as follows.

Definition 14 (hybrid trace) *A hybrid trace σ_h is a sequence of pairs σ_h: $\langle g_0, I_0\rangle$, $\langle g_1, I_1\rangle$, $\langle g_2, I_2\rangle$, $\ldots$, with $I_n \in \mathrm{Int}_{\mathbb{R}}$, $g_n : I_n \mapsto \Sigma$, for all $n \in \mathbb{N}$. The intervals can overlap at most at the endpoints, and they cover all $\mathbb{R}^+$: for all n,*

$$\sup I_n = \inf I_{n+1},$$

$$\bigcup_{n \in \mathbb{N}} I_n = \mathbb{R}^+.$$

Each function g_n assigns a state $g_n(t) \in \Sigma$ to each time $t \in I_n$. The discrete variables cannot change their value in an interval: for all $n \in \mathbb{R}$ and all $t_1, t_2 \in I_n$, $g_n(t_1)|_{\mathcal{V}_d} = g_n(t_2)|_{\mathcal{V}_d}$.

The value of variable x at time t of interval I_n is thus $g_n(t)(x)$. Again, we define admission only for closed traces, for simplicity.

Definition 15 (admission, hybrid traces) *A PTS S admits a trace σ_h: $\langle g_0, I_0\rangle$, $\langle g_1, I_1\rangle$, $\langle g_2, I_2\rangle$, $\ldots$, written $S \rhd \sigma_h$, if σ_h is closed and the following conditions are satisfied:*

1. *The phases are respected: for each $x \in \mathcal{V}_c$ and $n \in \mathbb{N}$, if $I_n^{\leftarrow} \neq I_n^{\rightarrow}$, there is a $\wp \in \mathcal{P}_x$ such that, for all $t \in I_n$:*

$$g_n(t) \in c_\wp,$$
$$f_\wp\, g_n(t) = \left.\frac{dg_n(u)(x)}{du}\right|_{u=t},$$

   where it is assumed that for $I_n^{\leftarrow} < t < I_n^{\rightarrow}$ the derivative $dg_n(u)(x)/du|_{u=t}$ exists, and for $t = I_n^{\leftarrow}$, $t = I_n^{\rightarrow}$, the left-hand and right-hand derivatives, respectively, exist.

2. *No immediate transition is skipped: for all n and $\tau \in \mathcal{T}_i$, $I_n^{\leftarrow} \leq t < I_n^{\rightarrow} \rightarrow g_n(t) \notin c_\tau$.*

3. *All discrete state changes are due to a transition: for all n, either $g_n(I_n^{\rightarrow}) = g_{n+1}(I_{n_1}^{\leftarrow})$ or $(g_n(I_n^{\rightarrow}), g_{n+1}(I_{n_1}^{\leftarrow})) \in \tau$ for some $\tau \in \mathcal{T}$. If such a τ is a delayed transition, we also require that it has been enabled for at least l_τ: for all $k \in \mathbb{N}$,*

$$k \leq n \wedge I_k^{\rightarrow} > I_n^{\rightarrow} - l_\tau \rightarrow g_k(I_k^{\rightarrow}) \in c_\tau.$$

4. *Delayed transitions never wait for longer than their maximum delay: for all $\tau \in \mathcal{T}_d$ and $n_1, n_2 \in \mathbb{N}$ with $n_2 \geq n_1$,*

$$I_{n_2}^{\rightarrow} - I_{n_1}^{\leftarrow} \leq u_\tau \vee \exists n_3 \Big[n_1 \leq n_3 \leq n_2 \wedge g_{n_3}(I_{n_3}^{\leftarrow}) \notin c_\tau \Big].$$

6.3 Discrete Semantics

The discrete semantics of hybrid systems is defined in terms of discrete traces, exactly as it was done for real-time systems in Definition 1. However, we do not define admission of discrete traces directly: we will define it through hybrid traces, using the translation functions.

6.4 Temporal Logic

Temporal logic is then defined for discrete and hybrid traces in the same way it was defined for discrete and continuous traces, respectively, for real-time systems. The logic corresponding to discrete traces is TL$_\mathrm{D}$, as before. The logic corresponding to hybrid traces will be called TL$_\mathrm{H}$, its satisfaction relation will be denoted with $\overset{\mathrm{H}}{\models}$ and its provability relation with $\overset{\mathrm{H}}{\vdash}$. We use a different name for TL$_\mathrm{H}$, as we do not wish to imply that TL$_\mathrm{C}$ and TL$_\mathrm{H}$ are the same. A deductive system for TL$_\mathrm{H}$ will be discussed later.

7 From Discrete to Continuous Reasoning

Refinement of discrete traces was defined in Definition 7. Refinement of hybrid traces is defined as follows.

Definition 16 (refinement, hybrid traces) *A hybrid trace σ_h: $\langle g_0, I_0\rangle$, $\langle g_1, I_1\rangle$, $\langle g_2, I_2\rangle$, ... is a refinement of σ'_h: $\langle g'_0, I'_0\rangle$, $\langle g'_1, I'_1\rangle$, $\langle g'_2, I'_2\rangle$, ... by the partitioning function μ, denoted $\sigma'_h \succeq^{\mu} \sigma_h$, if $I_i = \bigcup_{j\in\mu_i} I'_j$, and for every $i, j \in \mathbb{N}$ such that $j \in \mu_i$, it is $\forall t \in I'_j\left[g_j(t) \doteq g_i(t)\right]$.*

Sampling equivalence is then defined as before. The definition of the translation functions has to be modified, and we denote the new versions with $\Upsilon^{(h)}$, $\Omega^{(h)}$. In particular, a discrete trace no more encodes all the information required to reconstruct a hybrid trace: it contains the information about the state at the beginning and at the end of each closed interval, but it does not represent the evolution of the state in the interior of the interval. Therefore, to a single discrete trace correspond many hybrid ones that agree with the discrete one at the endpoints of the intervals.

Definition 17 ($\Upsilon^{(h)} : \sigma_d \mapsto \sigma_h$) *The translation function $\Upsilon^{(h)}$ associates to σ_d: $\langle s_0, t_0\rangle$, $\langle s_1, t_1\rangle$, $\langle s_2, t_2\rangle$, ... a set of closed hybrid traces $\Upsilon^{(h)}(\sigma_d)$, such that, for every σ_h: $\langle g_0, I_0\rangle$, $\langle g_1, I_1\rangle$, $\langle g_2, I_2\rangle$, ... $\in \Upsilon^{(h)}(\sigma_d)$, and for every n, it is $I_n^{\leftarrow} = t_n$, $I_n^{\rightarrow} = t_{n+1}$, $g_n(I_n^{\leftarrow}) = s_n$, $g_n(I_n^{\rightarrow}) = s_{n+1}$.*

Definition 18 ($\Omega^{(h)} : \sigma_h \mapsto \sigma_d$) *The translation function $\Omega^{(h)}$ associates to σ_h: $\langle g_0, I_0\rangle$, $\langle g_1, I_1\rangle$, $\langle g_2, I_2\rangle$, ... the discrete trace σ_d: $\langle s_0, t_0\rangle$, $\langle s_1, t_1\rangle$, $\langle s_2, t_2\rangle$, ... defined in the following way, for all $n \in \mathbb{N}$.*

1. *(a) If I_n is closed, $t_{2n} = I_n^{\leftarrow}$, $t_{2n+1} = I_n^{\rightarrow}$.*
   (b) If I_n is left open, $t_{2n} = t_{2n+1} = I_n^{\rightarrow}$.
   (c) If I_n is right open, $t_{2n} = t_{2n+1} = I_n^{\leftarrow}$.
   (d) If I_n is open, $t_{2n} = t_{2n+1} = (I_n^{\leftarrow} + I_n^{\rightarrow})/2$.
2. *$s_{2n} = g_n(I_n^{\leftarrow})$, $s_{2n+1} = g_n(I_n^{\rightarrow})$.*

A PTS S admits a discrete trace if the discrete trace describes a hybrid trace admitted by S. This is the implicit meaning of the definition given in [20].

Definition 19 (admission, discrete traces) *A PTS S admits a discrete trace σ_d, written $S \triangleright \sigma_d$, if there is a $\sigma_h \in \Upsilon^{(h)}(\sigma_d)$ such that $S \triangleright \sigma_h$.*

In defining finite variability for hybrid systems, it is essential to define it with respect to a given PTS, to constrain somehow the behavior of the continuous variables.

Definition 20 (HFV) *A formula ϕ is hybrid finite variability, or HFV, with respect to a PTS S if for every σ_h admitted by S and every $\mathcal{I}$, there exists a $\sigma'_h \succeq \sigma_h$ such that: $\mathcal{I}, \sigma'_c \models_{(i,t)} \psi \leftrightarrow \mathcal{I}, \sigma'_c \models_{(i,t')} \psi$ for all $\psi \in \mathrm{sb}(\phi)$.*

With these definitions, we can prove the corresponding of Theorem 4.

Theorem 7 (transfer of validity, hybrid case) *If $S \overset{\text{D}}{\models} \phi$ and ϕ is HFV with respect to S, then $S \overset{\text{H}}{\models} \phi$. If $\overset{\text{D}}{\models} \phi$ and ϕ is HFV with respect to S, then $S \overset{\text{H}}{\models} \phi$.*

Again, we present a sufficient condition for a formula to be HSFV with respect to a PTS S.

Definition 21 (simple age function) *We say that an age function $\Gamma(\phi)$ is simple with respect to a system S if its argument ϕ does not contain occurrences of continuous state variables of S.*

Definition 22 (syntactic finite variability, hybrid (HSFV)) *A formula is HSFV with respect to a PTS S if the phases of S are linear, and if the formula is constructed in the following inductive way.*

1. *If $u_1, \ldots, u_n$ are terms not containing T, Γ, or continuous variables, then $Pu_1 \ldots u_n$ is HSFV.*

2. *If $f(z_0, \ldots, z_n, v_1, \ldots, v_k)$ is a well-behaved function, then $f(b_0, \ldots, b_n, c_1, \ldots, c_k) = 0$, $f(b_0, \ldots, b_n, c_1, \ldots, c_k) > 0$ are HSFV formulas, provided $b_0, \ldots, b_k$ are constants of the logic or simple age functions, and $c_1, \ldots, c_k$ are variables of the logic, or constants different from T and from continuous state variables. We call this type of HSFV formulas T-atoms.*

3. *A formula constructed from HSFV formulas using propositional connectives or temporal operators is a HSFV formula.*

4. *If ϕ is a HSFV formula, and ξ does not occur in any T-atom of ϕ, then $\forall \xi \, \phi$ is a HSFV formula.*

Theorem 8 (HSFV implies HFV) *If ϕ is HSFV with respect to a PTS S, it is also HFV with respect to it. Therefore, the inference rules*

$$\frac{S \overset{\text{D}}{\vdash} \phi}{S \overset{\text{H}}{\vdash} \phi} \qquad \frac{S \overset{\text{D}}{\vdash}_0 \phi}{S \overset{\text{H}}{\vdash}_{(0,0)} \phi}$$

with the proviso that ϕ is HSFV with respect to S, are sound.

The restriction requiring the linearity of the phases is important, and cannot be lifted without being substituted by some other kind of condition insuring that the solutions of the differential equations are well-behaved in the sense of Definition 12.

A deductive system for TL$_\text{H}$. Since the definition of syntactic finite variability is now relative to a PTS, we need to modify slightly the deductive system proposed for TL$_\text{C}$. We take the same set of axioms, and all the inference rules listed in Table 4 apart from the last four, denoted by (§). Those four are replaced by the following rules:

$$\frac{\overset{\text{PTL}}{\vdash} \alpha[p_1, \ldots p_n]}{S \overset{\text{H}}{\vdash} \alpha[\phi_1, \ldots, \phi_n]} \qquad \frac{\overset{\text{PTL}}{\vdash}_0 \alpha[p_1, \ldots p_n]}{S \overset{\text{H}}{\vdash}_{(0,0)} \alpha[\phi_1, \ldots, \phi_n]} \qquad \frac{S \overset{\text{D}}{\vdash} \phi}{S \overset{\text{H}}{\vdash} \phi} \qquad \frac{S \overset{\text{D}}{\vdash}_0 \phi}{S \overset{\text{H}}{\vdash}_{(0,0)} \phi}$$

with the proviso that ϕ, ϕ_1, …, ϕ_n are HSFV with respect to S.

Acknowledgments. We wish to thank Nikølaj Bjorner, Amir Pnueli and Henny Sipma for their valuable comments and suggestions.

References

[1] R. Alur, T. Feder, and T.A. Henzinger. The benefits of relaxing punctuality. In *Proc. 10th ACM Symp. Princ. of Dist. Comp.*, pages 139–152, 1991.

[2] R. Alur and T.A. Henzinger. Logics and models of real time: a survey. In J.W. de Bakker, K. Huizing, W.-P. de Roever, and G. Rozenberg, editors, *Real Time: Theory in Practice*, Lecture Notes in Computer Science 600, pages 74–106. Springer-Verlag, 1992.

[3] J.P. Burgess. Axioms for tense logic I. "since" and "until". *Notre Dame journal of Formal Logic*, 23(4):367–374, October 1982.

[4] J.W. Garson. Quantification in modal logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 2, chapter 5. D. Reidel Publishing Company, 1984.

[5] E. Harel, O. Lichtenstein, and A. Pnueli. Explicit clock temporal logic. In *Proc. 5th IEEE Symp. Logic in Comp. Sci.*, pages 402–413, 1990.

[6] T. Henzinger, Z. Manna, and A. Pnueli. Temporal proof methodologies for real-time systems. In *Proc. 18th ACM Symp. Princ. of Prog. Lang.*, pages 353–366, 1991.

[7] T. Henzinger, Z. Manna, and A. Pnueli. Timed transition systems. In J.W. de Bakker, C. Huizing, W.P. de Roever, and G. Rozenberg, editors, *Proc. of the REX Workshop "Real-Time: Theory in Practice"*, volume 600 of *Lect. Notes in Comp. Sci.*, pages 226–251. Springer-Verlag, 1992.

[8] T. Henzinger, Z. Manna, and A. Pnueli. What good are digital clocks? In W. Kuich, editor, *Proc. 19th Int. Colloq. Aut. Lang. Prog.*, volume 623 of *Lect. Notes in Comp. Sci.*, pages 545–558. Springer-Verlag, 1992.

[9] T. Henzinger, Z. Manna, and A. Pnueli. Temporal proof methodologies for timed transition systems. *Inf. and Comp.*, 1994. To appear.

[10] T. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Information and Computation*, (111):193–244, 1994.

[11] Y. Kesten, Z. Manna, H. McGuire, and A. Pnueli. A decision algorithm for full propositional temporal logic. In *Computer Aided Verification, 5th International Workshop*, Lecture Notes in Computer Science. Springer-Verlag, 1993.

[12] Y. Kesten, Z. Manna, and A. Pnueli. Temporal verification of simulation and refinement. In *Proc. of the REX Workshop "A Decade of Concurrency"*, volume 803 of *Lect. Notes in Comp. Sci.*, pages 273–346. Springer-Verlag, 1994.

[13] Y. Kesten and A. Pnueli. TLR: Having your next and eating it too. Technical report, Weizmann Institute, 1992.

[14] L. Lamport. What good is temporal logic? In R.E.A. Mason, editor, *Proc. IFIP 9th World Congress*, pages 657–668. Elsevier Science Publishers (North-Holland), 1983.

[15] L. Lamport. The temporal logic of actions. Technical Report 79, DEC SRC, Palo Alto, CA, December 1991.

[16] L. Lamport. Verification and specification of concurrent programs. Technical report, DEC SRC, Palo Alto, CA, 1993.

[17] O. Lichtenstein, A. Pnueli, and L. Zuck. The glory of the past. In *Proc. Conf. Logics of Programs*, volume 193 of *Lect. Notes in Comp. Sci.*, pages 196–218. Springer-Verlag, 1985.

[18] O. Maler, Z. Manna, and A. Pnueli. From timed to hybrid systems. In J.W. de Bakker, C. Huizing, W.P. de Roever, and G. Rozenberg, editors, *Proc. of the REX Workshop "Real-Time: Theory in Practice"*, volume 600 of *Lect. Notes in Comp. Sci.*, pages 447–484. Springer-Verlag, 1992.

[19] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, New York, 1991.

[20] Z. Manna and A. Pnueli. Models for reactivity. *Acta Informatica*, 30:609–678, 1993.

[21] J. van Benthem. Correspondence theory. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 2, chapter 4. D. Reidel Publishing Company, 1984.